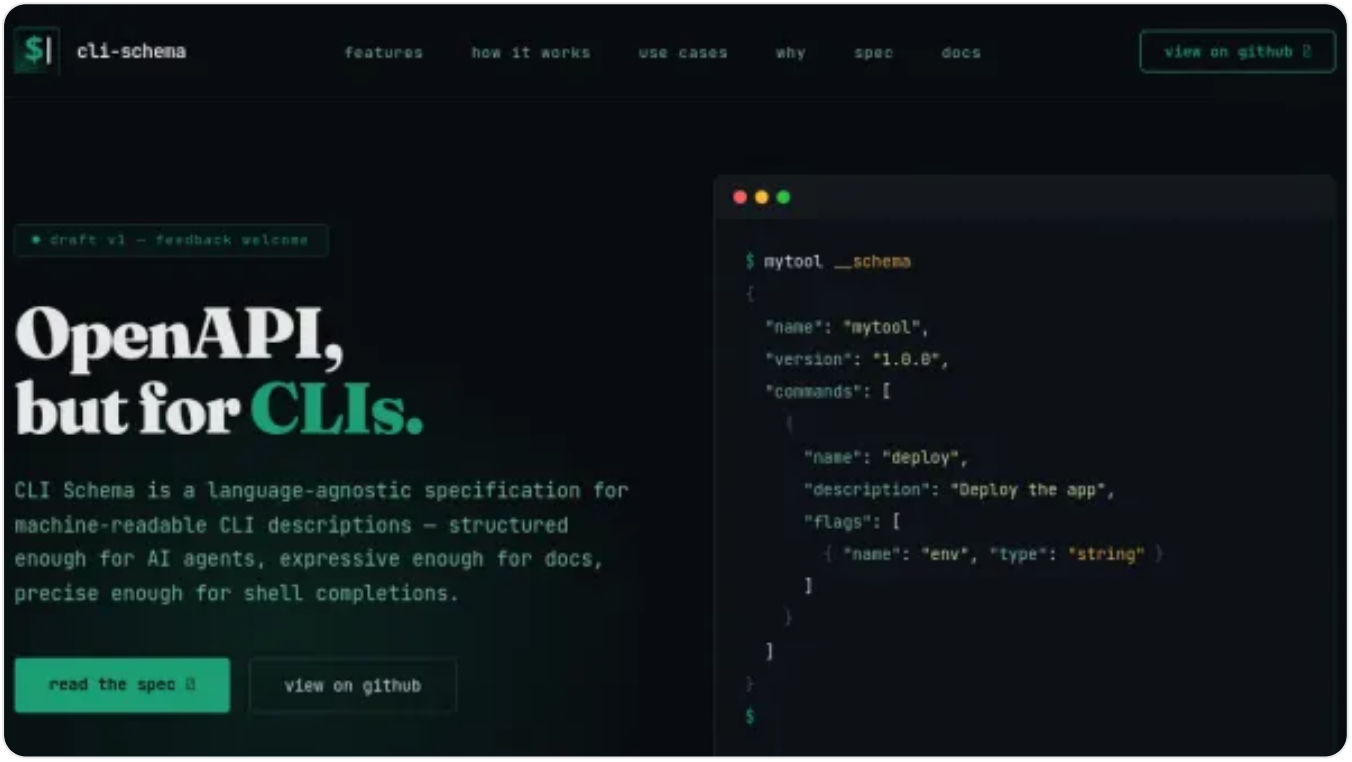
CS

CLI Schema

An open specification for machine-readable CLI descriptions.



About

CLI Schema is a CLI schema for machine-readable command-line descriptions. It defines an open, language-agnostic format that can be used to generate docs, shell completions, and AI-friendly interfaces from one source of truth. The project positions itself as “OpenAPI, but for CLIs,” with a draft v1 spec, a meta-schema for validation, and a CC0 dedication for broad reuse.

What is CLI Schema

CLI Schema is a CLI schema that describes a command-line tool in structured JSON so other software can read it without guessing from help text or source code. The project calls itself “OpenAPI, but for CLIs,” and the goal is to make one schema useful for docs

completions, validation, and agent tooling. It is language-agnostic, so the same CLI schema can represent tools written in Go, Rust, Python, Ruby, or other stacks.

The format is designed to be precise enough for shell completions and flexible enough for documentation and automation. A CLI schema can include a tool name, version, commands, nested subcommands, flags, short flag aliases, types, and required fields. The site shows an example with a `deploy` command and a string `env` flag, which makes the structure easy to read and easy to machine-parse.

Key features

CLI Schema focuses on turning one CLI schema into many downstream outputs.

- **AI-agent ready**: the schema is structured enough for LLMs and autonomous agents to reason about a CLI without training on its help output.
- **Docs generation**: reference docs can be produced directly from the schema, which reduces manual upkeep.
- **Shell completions**: the same source can power bash, zsh, fish, and PowerShell completions.
- **Language-agnostic**: it works regardless of implementation language, so the CLI schema is not tied to one runtime.
- **Validation support**: a JSON Schema meta-schema is included, so cli-schema documents can be checked against the spec.
- **Open licensing**: the specification is dedicated to the public domain under CC0 1.0 Universal.

A useful detail is that the project is not just about documentation. The CLI schema is intended as infrastructure for a wider ecosystem, including wrappers, agent interfaces, and editor integrations.

How to use it

You can expose a CLI schema in one of two ways, depending on how your tool is built.

- **Add a `_schema` command**

  Implement a `_schema` subcommand in your CLI

This lets tools discover the schema at runtime.

Ship a sidecar file

Place a file such as `mytool.cli-schema.json` next to the binary.

This approach requires no runtime changes.

Let tooling consume the schema

Any CLI schema-aware tool can then generate docs, completions, type-safe wrappers, or agent interfaces.

The same schema can also be validated against the included meta-schema.

The site also shows a GitHub CLI example with nested commands and a required `title` flag, which illustrates how a CLI schema can represent real command trees rather than only flat command lists.

Who it is for

CLI Schema is useful for teams and projects that want a single authoritative definition of a CLI schema instead of maintaining separate docs and completion scripts.

- **CLI maintainers** who want accurate generated reference material.
- **Tooling authors** building completion engines, wrappers, or parsers.
- **AI and agent developers** who need a structured description of commands and flags.
- **Developer experience teams** looking to surface inline docs and validation in editors.
- **Open-source projects** that want a reusable, license-free specification.

Because the format is open and CC0 licensed, it is attractive for both commercial and community projects that want to adopt a shared CLI schema without licensing friction.

What to know before you use it

CLI Schema is still a draft v1 specification, so it is best treated as an emerging standard rather than a finalized universal convention. The site explicitly invites feedback, contributions, and implementations.

A few caveats matter in practice:

It is a specification, not a hosted SaaS product, so there is no indication of pricing, accounts, or managed infrastructure.

The project is focused on describing CLIs; it does not replace your actual command implementation.

Generated docs and completions are only as accurate as the schema you provide, so stale input will produce stale output.

The included meta-schema can validate structure, but it cannot guarantee that your command behavior matches the schema.

For authoritative behavior, the CLI's own runtime output and `--help` text still matter, especially if the schema has not been updated after a release.

If you are adopting CLI schema for a production tool, keep the specification and the implementation in sync, and verify that your generated outputs match the live CLI. The better your CLI schema, the more reliable your docs, completions, and agent integrations will be.

FAQ

Is CLI Schema free to use?

How do I expose CLI Schema in my tool?

What problem does CLI Schema solve?

What can other tools do with a CLI Schema document?

How do I expose my CLI schema?

Which platforms or shells does it support?

Does CLI Schema work with any programming language?

Who is CLI Schema best for?

Can I validate a CLI schema document?

Sponsored

Belay – Awa... Featured
Developer Tools

Stays awake while your agents work.

Free # Developer Tools

View ↗

Partly Featured
Finance

Partly is a mobile app that lets friends quickly scan receipts, split any bill fairl...

Free # Finance

View ↗

Visit website ↗

cli-schema

Pricing	Free
Category	AI
Platform	Web, API
Launched	Aug 2026
Updated	17 days ago
Verified	1 months ago

TAGS

ai-agents # specification # documentation # shell-completions # json-schema # cli # open source

Dir Hub

A curated directory of small, well-made apps. Browse by category or search to find the right one in fewer clicks.

DISCOVER

Home

Pricing

Submit your app

COMPANY

About

Contact

LEGAL

Privacy

Terms

© 2026 Dir Hub

Curated indie apps

CS CLI Schema